(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 197 967 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
17.04.2002 Bulletin 2002/16

(51) Int Cl.7: G11B 20/14, G11B 20/18

(21) Application number: 01307489.3

(22) Date of filing: 03.09.2001

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 11.10.2000 KR 2000059840

(71) Applicant: SAMSUNG ELECTRONICS CO., LTD.
Suwon-City, Kyungki-do (KR)

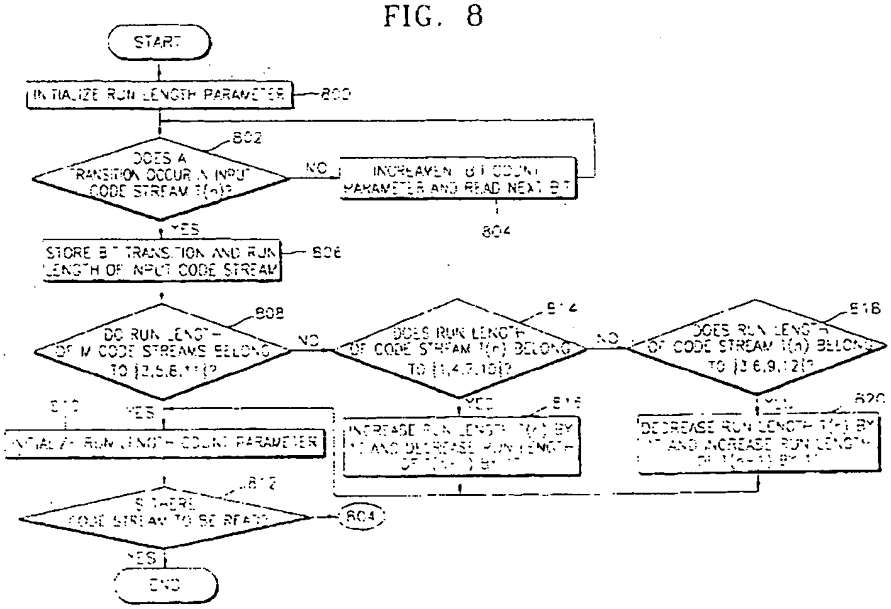
(72) Inventors:
- Shim, Jae-seong
Gwangjin-gu, Seoul (KR)
- Lee, Kyung-geun, 122-1002 Sibeom Hanshin
Seongnam-si, Gyeonggi-do (KR)
- Kim, Ki-hyun
Youngsan-gu, Seoul (KR)
- Park, Hyun-soo
Seocho-gu, Seoul (KR)

(74) Representative: Chugg, David John et al
Appleyard Lees,
15 Clare Road
Halifax, West Yorkshire HX1 2HY (GB)

(54) **Data modulation and correction methods**

(57) A data modulation method resistant to channel distortion and a method for correcting error in data coded by the modulation method are provided. The data modulation method uses a run length limited (RLL) modulation code applied to write data to an optical storage medium, the RLL modulation code being expressed as RLL (*d, k, m, n, s*) with *s* = 2 or greater, where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and *s* is the space length between codewords. The data modulation and correction methods can reduce a redundancy in physical address data written to an optical storage medium with improved detection performance resistant to disturbance.

FIG. 8



EP 1 197 967 A2

Description

[0001] The present invention relates to data modulation and correction methods, and more particularly, to a code modulation method for modulating data to be resistant to channel distortion and a method for correcting errors in detecting the data modulated by the modulation method.

[0002] With the increased recording density of optical storage media, track pitches of digital versatile disc random access memories (DVD-RAMs), recordable DVD (DVD-R), or DVD rewritable (DVD-RW) become narrow. Accordingly, introduction of crosstalk or noise from neighbor tracks and the amount of inter-symbol Interference (ISI) from neighbor pits are increasing. This disturbance increases jitter of a radio frequency (RF) signal, particularly when an optical storage medium whose physical address is recorded as pits is accessed.

[0003] Run length limited (RLL) codes having limited maximum and minimum constraints are common data modulation codes for optical storage media. The minimum run length, which is denoted by parameter "d", greatly affects accuracy in detecting pits (or lands) in an optical disc and the recording density of that code. The maximum run length, which is denoted by parameter "k", is associated with code efficiency and code strategy. Common RLL codes include consecutive run lengths between the minimum and maximum run lengths. For example, for eight-to-fourteen modulation plus (EFM+) codes compatible with DVD family recording media with d=2 and k=10, 4T, 5T, ..., and 10T (where T is a reproduction clock) codes, excluding a sync code, exist between a minimum pit length of 3T and a maximum pit length of 11T. Figure 1 is a histogram of the run length distribution of the EFM+ codes. However, when jitter occurs in an RF signal being reproduced, due to any distortion in a channel, the 1T space codes of Figure 1 increase the likelihood of error occurring at a deviation of ∀0.5T.

[0004] Figure 2 is a table of the physical address format in the header field of a general DVD-RAM. For the general DVD-RAM, the same address data are written twice to the header field for high detection performance. The header field includes variable frequency oscillator (VFC1) data for a phase locked loop (PLL), address mark (AM) data, physical identification data (PID1), ID error detection data (IED1), and postamble data (PA1) for IED1 demodulation. VFO2, AM, PID2, IED2, and PA2, which have the same functions as those above, are also written to the header field. However, even though detection performance is improved, writing the physical address data based upon this RLL modulation code causes a redundancy problem.

[0005] With a view to solve or reduce the above-described problems, it is an aim of embodiments of the present invention to provide a data modulation method which can modulate data to be resistant to disturbance, and a method for correcting errors occurring in reading the data written by the modulation method.

[0006] According to an aspect of the present invention, there is provided a data modulation method using a run length limited (RLL) modulation code applied to write data to an optical storage medium, the RLL modulation code being expressed as RLL (d, k, m, n, s) with s = 2 or greater, where d is minimum run length, k is maximum run length, m is the data bit length before modulation, n is the codeword bit length after modulation, and s is the space length between codewords.

[0007] It is preferable that, in the RLL modulation code, the run length is one of 2, 5, 8, and 11, the number of lead zeros of a codeword is one of 1, 4, and 7, and the number of end zeros of a codeword is one of 1 and 4.

[0008] It is preferable that the RLL modulation code is applied to write address data to a physical sector of the optical storage medium, the address data being required for data access.

[0009] It is preferable that the RLL modulation code is pre-formed as pits in the optical storage medium to write and store importance information including copyrights.

[0010] It is preferable that the data modulation method comprises discarding codewords having a maximum run length from codewords modulated with the RLL modulation code for the purpose of generating a read clock signal and a synchronization clock signal.

[0011] It is preferable that the data modulation method comprises discarding codewords having a codeword digital sum (CDS) whose absolute value is relatively great, from codewords modulated with the RLL modulation code.

[0012] According to another aspect of the present invention, there is provided a data modulation method using a run length limited (RLL) modulation code applied to write data to an optical storage medium, the RLL modulation code being expressed as RLL (d, k, m, n, s), where d is minimum run length, k is maximum run length, m is the data bit length before modulation, n is the codeword bit length after modulation, and s is the space length between codewords, the data modulation method comprising: generating codewords satisfying an RLL (2,11,8,27,3) code; and removing codewords having a run length of 12T from the generated codewords, and creating a codeword table of the remaining codewords, where T is a reproduction clock signal.

[0013] It is preferable that the data modulation method further comprises: discarding codewords having a codeword digital sum (CDS) value whose absolute value is relatively great, from the codeword table; and arranging the remaining codewords in ascending order of the absolute value of the CDS for the purpose of reducing a direct current (DC) component of the codewords.

[0014] In another embodiment of the data correction method according to the present invention, there is provided a method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL)

modulation code expressed as RLL (*d, k, m, n, s*), where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and *s* is the space length between codewords, the method comprising: (a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions in the code sequences, and if an N-th pit that does not comply with prescribed RLL coding rules is detected, checking the run lengths of following M pits; and (b) if the run length of any of the following M pits does not comply with the prescribed RLL coding rules, correcting the run length of the N-th pit according to the RLL coding rules.

[0015] It is preferable that step (b) comprises: if the run length of the N-th pit is s/2 less than a run length of the RLL code, correcting the N-th pit by adding s/2 to the run length of the N-th pit; and correcting the (N+1)th pit by subtracting s/2 from the run length of the (N+1)th pit.

[0016] It is preferable that step (b) comprises: if the run length of the N-th pit is s/2 greater than a run length of the RLL code, correcting the N-th pit by subtracting s/2 from the run length of the N-th pit; and correcting the (N+1)th pit by adding s/2 to the run length of the (N+1)th pit.

[0017] In the data correction method above, when s=3, step (b) preferably comprises: if the run length of the N-th pit is 1 less than a run length of the RLL code, correcting the N-th pit by adding 1 to the run length of the N-th pit; and correcting the (N+1)th pit by subtracting 1 from the run length of the (N+1)th pit.

[0018] When s=3, it is preferable that step (b) comprises: if the run length of the N-th pit is 1 greater than a run length of the RLL code, correcting the N-th pit by subtracting 1 from the run length of the N-th pit; and correcting the (N+1)th pit by adding 1 to the run length of the (N+1)th pit.

[0019] In still another embodiment of the data correction method according to the present invention, there is provided a method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL) modulation code expressed as RLL (*d, k, m, n, s*) with *d*=1 and *s*=3, where d is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and *s* is the space length between codewords, the method comprising: (a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions, determining whether the counted run lengths comply with prescribed RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length of i_1=2, and if an N-th pit which does not equal to any of the run lengths i_n is detected, checking the run lengths of the following M pits; and (b) if the run length of at least one of the M pits is not equal to the run lengths i_n, correcting the run length of the N-th pit according to the prescribed RLL modulation rules.

[0020] It is preferable that step (b) comprises: (b1) correcting the run length of the N-th pit by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit; and (b2) correcting the run length of the (N+1)th pit by subtracting from or adding to the run length of the N-th pit the value which is added or subtracted in step (b1).

[0021] In yet still another embodiment of the data correction method according to the present invention, there is provided a method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL) modulation code expressed as RLL (*d, k, m, n, s*) with *d*=1 and *s*=3, where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and *s* is the space length between codewords, the method comprising: (a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions, determining whether the counted run lengths comply with prescribed RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length of i_1=1, and if an N-th pit which does not equal any of the run lengths i_n is detected, checking the run lengths of the following M pits; and (b) if the run length of at least one of the M pits is not equal to the run lengths i_n, correcting the run length of the N-th pit to comply with the prescribed RLL modulation rules.

[0022] It is preferable that step (b) comprises: (b1) correcting the run length of the N-th pit by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit; and (b2) correcting the run length of the (N+1)th pit by subtracting from or adding to the run length of the N-th pit the value which is added or subtracted in step (b1).

[0023] For a better understanding of the invention, and to show how embodiments of the same may be carried into effect, reference will now be made, by way of example, to the accompanying diagrammatic drawings in which:

Figure 1 is a histogram of the run length distribution of EFM+ codes;

Figure 2 is a table of the physical address format in the header field of a general digital versatile disc random access memory (DVD-RAM) ;

Figure 3 is a histogram of the distribution of a codestream modulated by a data modulation method according to an embodiment of the present invention;

Figure 4 is a table of the codewords generated according to prescribed rules and classified by a codeword digital sum (CDS);

Figures 5A and 5B are tables of extended space run length limited (ES RLL) (2, 11, 8, 27, 3) modulation codewords from which codewords having a maximum pit length of 12T are discarded;

Figures 6A and 6B are tables of ES RLL (2, 11, 8, 27, 3) modulation codewords which have smaller CDS;

Figure 7 is a flowchart illustrating a method of correcting errors caused by jitter when data written to an optical storage medium with the ES RLL modulation code according to embodiment of the present invention are read;

Figure 8 is a flowchart illustrating an error correction method in detecting codewords modulated by an RLL (2, 11, 8, 27, 3) code;

Figure 9 shows the structure of the physical address data written to an optical storage medium for a conventional RLL code (a) and the ES RLL code according to embodiments of the present invention (b);

Figure 10 is a table showing the data format of the physical address sector of an optical storage medium to which the ES RLL code according to embodiments of the present invention is applied;

Figure 11A is a histogram of the ES RLL code stream according to embodiments of the present invention digitized using a conventional level slicer;

Figure 11B is a histogram of the ES RLL code stream according to embodiments of the present invention digitized using an SRC detector;

Figure 12 is a graph showing code stream detection performance with respect to circuit noise for the conventional method and that of embodiments of the present invention;

Figure 13 is a graph showing code stream detection performance with respect to asymmetry for the conventional method and that of embodiments of the present invention; and

Figure 14 is a graph showing code stream detection performance with respect to tangential skew of an optical storage medium for the conventional method and that of embodiments of the present invention.

[0024] A data modulation method according to the teachings of the present invention writes data to an optical storage medium using a modulation code spaced 2T or greater between pits, thereby reducing detection errors. Figure 3 is an exemplary histogram of the distribution of the codestream generated by a data modulation method according to an embodiment of the present invention. Use of codes spaced 2T or 3T between pits further improve data detection performance, compared to 1T spaced codes, for the same amount of jitter. Although the 2T or 3T spaced codes are disadvantageous in terms of recording density due to the increased code length, these codes are effective for important information such as physical address data written to a disc, in terms of improved detection performance with reduced errors in the presence of jitter.

[0025] The modulation method according to the present invention relates to generation of RLL(d, k, m, n, s) modulation codes with $s = 2$ or greater for improved detection performance, where d is minimum run length, k is maximum run length, m is the data bit length before modulation, n is the codeword bit length after modulation, and s is the space length between codewords Here, the run length means the number of consecutive "0"s between neighboring logical "1"s in binary code sequences.

[0026] RLL (2, 11, 8, 27, 3) codewords as shown in the histogram of Figure 3, which are generated with s=3, can be obtained according to the following rules:

(1) Run length ∈ {2, 5, 8, 11}

(2) Lead zero of a codeword ∈ {1, 4, 7}, and

(3) End zero of a codeword E {1, 4}.

[0027] According to Rule (1), the length of pits is one of 3T, 6T, 9T, or 12T. Rules (2) and (3) are established such that Rule (1) is satisfied when codewords are connected. For example, if the trailing end of a codeword has one zero, and the following codeword has one zero at its leading end, the run length of the two codewords becomes 2, satisfying the minimum run length defined by Rule (1). If the trailing end of a codeword has 4 zeros, and the following codeword has four zeros at its front end, the run length of the two codewords becomes 11, satisfying the maximum run length of Rule (1). According to this modulation, space between pits is extended by 3T in the present embodiment, so this modulation code is called "extended space modulation (ES RLL) code". The number of codewords satisfying Rules (1) through (3) is 298 in total, and the number of codewords having a maximum pit length of 12T is 63.

[0028] Figure 4 is a table of the codewords generated to comply with Rules (1) through (3), which are classified by a codeword digital sum (CDS). CDS refers to the result of summation of binary bits "1" and "0" in each codeword, wherein bits "0" are replaced by "-1" for the sum-

mation.

[0029] Figures 5A and 5B are tables of ES RLL (2. 11, 8, 27, 3) modulation codewords from which the 63 codewords having a maximum pit length of 12T are discarded to avoid a problem in the reproduction of a clock signal or of sync mismatching, wherein the discarded 63 codewords may include two pits having the maximum length of 12T.

[0030] Figures 6A and 6B are tables of ES RLL (2. 11, 8, 27, 3) modulation codewords which have a smaller CDS. In other words, codewords having a relatively greater CDS are removed from the modulation codewords of Figures 6A and 6B so as to reduce the running digital sum (RDS) for direct current (DC) suppression control to the codewords.

[0031] The ES RLL modulation codes are pre-formed as pits in the physical address sector (header field) of an optical storage medium, to write physical address data for indicating data locations or other important information associated with, for example, copyrights.

[0032] An error correction method applied in detecting codewords modulated by the method described above will now be described.

[0033] Figure 7 is a flowchart illustrating a method of correcting errors caused by jitter when data written to an optical storage medium with the ES RLL modulation code described above is read. In correcting errors in the data written with an RLL (*d, k, m, n,* s) code and read from an optical storage medium, first the run lengths of code sequences read from the optical storage medium are counted by detecting bit transitions therein. If an Nth-pit (or mark) which does not comply with the RLL coding rules is detected, the run lengths of the following M pits are checked (step 700). If the run length of any of the following M pits does not comply with the RLL coding rules, the run length of the N-th pit is corrected according to the RLL coding rules (step 710).

[0034] In step 710 of Figure 7, if the run length of the N-th pit is s/2 less than the run length of the RLL code, the N-th pit is corrected by adding s/2 to the run length of the N-th pit, and the (N+1)th pit is corrected by subtracting s/2 from the run length of the (N+1)th pit. If the run length of the N-th pit is s/2 greater than the run length of RLL (*d, k, s*), the N-th pit is corrected by subtracting s/2 from the run length of the N-th pit, and the (N+1)th pit is corrected by adding s/2 to the run length of the (N+1)th pit.

[0035] For an RLL (*d, k, s*) code with s=3, in step 710 of Figure 7, if the run length of the N-th pit is 1 less than the run length of the RLL code, the N-th pit is corrected by adding 1 to the run length of the N-th pit, and the (N+1)th pit is corrected by subtracting 1 from the run length of the (N+1)th pit. If the run length of the N-th pit is 1 greater than the run length of the RLL code, the N-th pit is corrected by subtracting 1 from the run length of the N-th pit, and the (N+1) th pit is corrected by adding 1 to the run length of the (N+1) th pit.

[0036] In correcting data read from an optical storage medium after having been written in an RLL (*d, k, m, n, s)* code with *d*=1 and *s*=3, where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation. and s is the space length between codewords, the run lengths of code sequences read from the optical storage medium are counted by detecting bit transitions therein. Next, it is determined whether the counted run lengths comply with the RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length i_1=2. If an N-th pit which does not equal any of the run lengths i_n is detected, the run lengths of the following M pits are checked. If the run length of at least one of the M pits is not equal to the run lengths i_n, the run length of the N-th pit is corrected to comply with the RLL modulation rules.

[0037] In particular, the run length of the N-th pit is corrected by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit, and is followed by correction in the run length of the (N+1)th pit. If a value is added to or subtracted from the run length of the N-th pit, the run length of the (N+1)th pit is corrected by subtracting the added value from or adding the subtracted value to the same.

[0038] As another embodiment, in correcting data read from an optical storage medium which was written in an RLL (*d, k, m, n, s)* code with *d*=1 and *s*=3, where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and s is the space length between codewords, the run lengths of code sequences read from the optical storage medium are counted by detecting bit transitions therein. Next, it is determined whether the counted run lengths comply with the RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length of i_1=1. If an N-th pit which does not equal to the run lengths i_n is detected, the run lengths of the following M pits are checked. If the run length of at least one of the M pits is not equal to the run lengths i_n, the run length of the N-th pit is corrected to comply with the RLL modulation rules.

[0039] In particular, the run length of the N-th pit is corrected by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit, and is followed by correction in the run length of the (N+1)th pit. If a value is added to or subtracted from the run length of the N-th pit, the run length of the (N+1)th pit is corrected by subtracting the added value therefrom or adding the subtracted value thereto.

[0040] In the data correction methods described above, if the run lengths of the M pits comply with the RLL coding rules, even if an error is detected in the N-

th pit, the run length of the N-th pit is not corrected to prevent error propagation.

[0041] Figure 8 is a flowchart illustrating an error correction method in detecting a RLL (2, 11, 8, 27, 3) modulated code. When a code stream T(n) is input, a run length count parameter is initialized (step 800). Next, it is determined whether a transition occurs in a current bit of the input code stream (step 802). If no transition occurs in the current bit, a bit count parameter is incremented to read a next bit (step 804), and step 802 is performed again. If a transition is detected in step 802, the bit transition position and the run length of the input code stream T(n) are stored (step 806). The above steps are repeated for M code streams T(n+1), T(n+2), T (n+3), and T(n+4), following the code stream T(n), where M=4 in the present embodiment. Next, it is checked whether the run length of each of the M code streams belong to {2, 5, 8, 11} (step 808). In other words, it is determined whether the M code streams have 3T, 6T, 9T, or 12T signal. If all the run lengths of the code streams T(n+1), T(n+2), T(n+3), and T(n+4) belong to {2, 5, 8, 11}, the run length count parameter is initialized (step 810). Next, it is determined whether there is a code stream to be read (step 812). If there is no code stream to be read, the process stops. If there is a code stream to be read, the process goes to step 802. If it is determined in step 808 that the run length of at least one of the M code streams T(n+1), T(n+2), T(n+3), and T(n+4) does not belong to {2, 5, 8, 11}, it is determined whether the run length of the code stream T(n) stored in step 806 belongs to {1, 4, 7, 10} (step 814). If the run length of the code stream T(n) stored in step 806 belongs to {1, 4, 7, 10}, it means that the code stream T(n) has a run length 1T shorter than the desired run length with a pit signal of 2T, 5T, 8T, or 11T. The pit signal 2T, 5T, 8T, or 11T is corrected into 3T, 6T, 9T, or 12T, respectively, by increasing the run length of the code stream T(n). Subsequently, the code stream T(n+1) following the code stream T(n) is corrected by subtracting 1T from the run length of the code stream T(n+1) (step 816). If it is determined in step 814 that the run length of the code stream T(n) does not belong to {1, 4, 7, 10}, it is determined whether the run length of the code stream T(n) belongs to {3; 6, 9, 12} (step 818). If the run length of the code stream T(n) belongs to {3, 6, 9, 12}, it means that the code stream T(n) has a run length 1T longer than the desired run length with a pit signal of 4T, 7T, 10T, or 13T. The pit signal 4T, 7T, 10T, or 13T is corrected into 3T, 6T, 9T, or 12T, respectively, by reducing the run length of the code stream T(n). Subsequently, the code stream T(n+1) following the code stream T(n) is corrected by adding 1T to the run length of the code stream T (n+1) (step 820).

[0042] The data correction according to embodiments of the present invention is referred to as "SRC detection (Slide Run length Compensation looking forward pits)" because the correction is performed by checking pits for compliance with the applied coding rules.

[0043] Figure 9 shows the format of the physical address data written to an optical storage medium for a conventional RLL code (a) and the ES RLL code according to the present invention (b). Figure 10 is a table showing the data format of the physical sector of an optical storage medium to which the ES RLL code according to the present invention is applied. As shown in Figure 10, the physical address data includes variable frequency oscillator (VFO) data for a phase locked loop (PLL), address mark (AM) data, physical identification data (PID), and ID error detection data (IED). Unlike a conventional physical address data format having 1004 bits in total, the constituent data are written once with a reduced total bit number of 783. As a result, there is an effect of 22% redundancy reduction in the physical address sector.

[0044] Figure 11A is a histogram of the ES RLL code stream according to embodiments of the present invention digitized using a conventional level slicer. As shown in Figure 11A, pits of the code stream range from 2T to 12T in length with 1T space. 192 pits of 2T, 10135 pits of 3T, 343 pits of 3T, 141 pits of 5T, 5621 pits of 6T, 3 pits of 7T, 18 pits of 8T, 2892 pits of 9T, 7 pits of 10T, 8 pits of 11T, and 589 pits of 12T are detected. The pits excluding 3T, 6T, 9T, and 12T pits correspond to errors.

[0045] Figure 11B is a histogram of the ES RLL code stream according to embodiments of the present invention digitized using an SRC detector. As shown in Figure 11B, 10671 pits of 3T, 5763 pits of 6T, 1917 pits of 9T, and 595 pits of 12T are detected. The errors shown in Figure 11A disappear.

[0046] Figures 12 through 14 shows the difference in detection efficiency between different coding methods. Figure 12 is a graph showing code stream detection performance with respect to circuit noise for the conventional method and embodiments of the present invention. As shown in Figure 12, compared to the conventional method in which a level slicer is applied to the RLL code stream, when the SRC detection method is applied to the ES RLL code stream, detection performance is 5dB or greater improved.

[0047] Figure 13 is a graph showing code stream detection performance with respect to asymmetry for embodiments of the conventional method and the present invention. As shown in Figure 13, when the ES RLL code stream is detected using the SRC detector, an asymmetry margin is about 5% greater than the conventional method.

[0048] Figure 14 is a graph showing code stream detection performance with respect to the tangential skew of an optical storage medium for the conventional method and that embodiments of the present invention. As shown in Figure 14, when the ES RLL code stream is detected using the SRC detector, a tilt margin at an error rate of 10^{-4} is about 0.5 degrees greater than the conventional method.

[0049] As described above, the data modulation and correction methods according to the teachings of the

present invention can reduce a redundancy in physical address data written to an optical storage medium with improved detection performance tolerant to disturbance.

[0050] While this invention has been particularly shown and described with reference to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the scope of the invention as defined by the appended claims.

[0051] The reader's attention is directed to all papers and documents which are filed concurrently with or previous to this specification in connection with this application and which are open to public inspection with this specification, and the contents of all such papers and documents are incorporated herein by reference.

[0052] All of the features disclosed in this specification (including any accompanying claims, abstract and drawings), and/or all of the steps of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or steps are mutually exclusive.

[0053] Each feature disclosed in this specification (including any accompanying claims, abstract and drawings), may be replaced by alternative features serving the same, equivalent or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise, each feature disclosed is one example only of a generic series of equivalent or similar features.

[0054] The invention is not restricted to the details of the foregoing embodiment(s). The invention extend to any novel one, or any novel combination, of the features disclosed in this specification (including any accompanying claims, abstract and drawings), or to any novel one, or any novel combination, of the steps of any method or process so disclosed.

Claims

1. A data modulation method using a run length limited (RLL) modulation code applied to write data to an optical storage medium, the RLL modulation code being expressed as RLL (*d, k, m, n, s*) with *s* = 2 or greater, where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and s is the space length between codewords.

2. The data modulation method of claim 1, wherein, in the RLL modulation code, the run length is one of 2, 5, 8, and 11, the number of lead zeros of a codeword is one of 1, 4, and 7, and the number of end zeros of a codeword is one of 1 and 4.

3. The data modulation method of claim 1 or 2, wherein the RLL modulation code is applied to write address data to a physical sector of the optical storage medium, the address data being required for data access.

4. The data modulation method of claim 1, 2 or 3, wherein the RLL modulation code is pre-formed as pits in the optical storage medium to write and store importance information including copyrights.

5. The data modulation method of claim 1, 2, 3 or 4, comprising discarding codewords having a maximum run length from codewords modulated with the RLL modulation code for the purpose of generating a read clock signal and a synchronization clock signal.

6. The data modulation method of any preceding claim, comprising discarding codewords having a codeword digital sum (CDS) whose absolute value is relatively great, from codewords modulated with the RLL modulation code.

7. A data modulation method using a run length limited (RLL) modulation code applied to write data to an optical storage medium, the RLL modulation code being expressed as RLL (*d, k, m, n, s*), where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation, and s is the space length between codewords, the data modulation method comprising:

   generating codewords satisfying an RLL (2, 11, 8, 27, 3) code; and

   removing codewords having a run length of 12T from the generated codewords, and creating a codeword table of the remaining codewords, where T is a reproduction clock signal.

8. The data modulation method of claim 7, further comprising:

   discarding codewords having a codeword digital sum (CDS) value whose absolute value is relatively great, from the codeword table; and

   arranging the remaining codewords in ascending order of the absolute value of the CDS for the purpose of reducing a direct current (DC) component of the codewords.

9. A method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL) modulation code expressed as RLL (*d, k, m, n, s*), where *d* is minimum run length, *k* is maximum run length, *m* is the data bit length before modulation, *n* is the codeword bit length after modulation,

and s is the space length between codewords, the method comprising:

(a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions in the code sequences, and if an N-th pit that does not comply with prescribed RLL coding rules is detected, checking the run lengths of following M pits; and

(b) if the run length of any of the following M pits does not comply with the prescribed RLL coding rules, correcting the run length of the N-th pit according to the RLL coding rules.

10. The method of claim 9, wherein step (b) comprises:

if the run length of the N-th pit is s/2 less than a run length of the RLL code, correcting the N-th pit by adding s/2 to the run length of the N-th pit; and

correcting the (N+1)th pit by subtracting s/2 from the run length of the (N+1)th pit.

11. The method of claim 9 or 10, wherein step (b) comprises:

if the run length of the N-th pit is s/2 greater than a run length of the RLL code, correcting the N-th pit by subtracting s/2 from the run length of the N-th pit; and

correcting the (N+1)th pit by adding s/2 to the run length of the (N+1)th pit.

12. The method of claim 9, 10 or 11, wherein, when s=3, step (b) comprises:

if the run length of the N-th pit is 1 less than a run length of the RLL code, correcting the N-th pit by adding 1 to the run length of the N-th pit; and

correcting the (N+1)th pit by subtracting 1 from the run length of the (N+1)th pit.

13. The method of claim 9, 10, 11 or 12, wherein, when s=3, step (b) comprises:

if the run length of the N-th pit is 1 greater than a run length of the RLL code, correcting the N-th pit by subtracting 1 from the run length of the N-th pit; and

correcting the (N+1)th pit by adding 1 to the run length of the (N+1)th pit.

14. A method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL) modulation code expressed as RLL (d, k, m, n, s) with d=1 and s=3, where d is minimum run length, k is maximum run length, m is the data bit length before modulation, n is the codeword bit length after modulation, and s is the space length between codewords, the method comprising:

(a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions, determining whether the counted run lengths comply with prescribed RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length of i_1=2, and if an N-th pit which does not equal to any of the run lengths i_n is detected, checking the run lengths of the following M pits; and

(b) if the run length of at least one of the M pits is not equal to the run lengths i_n, correcting the run length of the N-th pit according to the prescribed RLL modulation rules.

15. The method of claim 14, wherein step (b) comprises:

(b1) correcting the run length of the N-th pit by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit; and

(b2) correcting the run length of the (N+1)th pit by subtracting from or adding to the run length of the N-th pit the value which is added or subtracted in step (b1).

16. A method for correcting data read from an optical storage medium after the data has been written to the optical storage medium with a run length limited (RLL) modulation code expressed as RLL (d, k, m, n, s) with d=1 and s=3, where d is minimum run length, k is maximum run length, m is the data bit length before modulation, n is the codeword bit length after modulation, and s is the space length between codewords, the method comprising:

(a) counting the run lengths of code sequences read from the optical storage medium by detecting bit transitions, determining whether the counted run lengths comply with prescribed RLL modulation rules, where the run lengths are expressed as $i_{n+1} = i_n+3$ (n=1, 2, ...) with an initial run length of i_1=1, and if an N-th pit

which does not equal any of the run lengths i_n is detected, checking the run lengths of the following M pits; and

(b) if the run length of at least one of the M pits is not equal to the run lengths i_n, correcting the run length of the N-th pit to comply with the prescribed RLL modulation rules.

17. The method of claim 16, wherein step (b) comprises:

(b1) correcting the run length of the N-th pit by adding a value to or subtracting a value from the run length of the N-th pit such that the run length of the N-th pit is equal to the run length i_n having an absolute value closest to that of the N-th pit; and

(b2) correcting the run length of the (N+1)th pit by subtracting from or adding to the run length of the N-th pit the value which is added or subtracted in step (b1).

FIG. 1

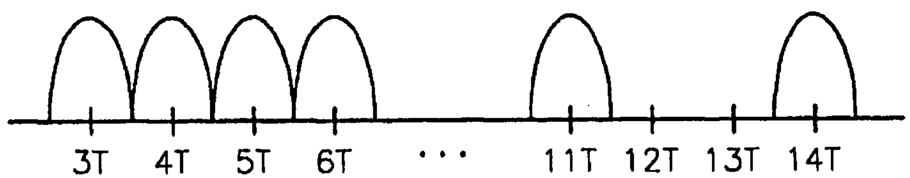


FIG. 2

SEGMENT	VFO1	AM	PID1	IED1	PAI	VFO2	AM	PID2	IED2	PA2
BYTES	36	3	4	2	1	8	3	4	2	1
BITS	576	48	54	32	16	128	48	54	32	16
BITS IN TOTAL	1004									

FIG. 3

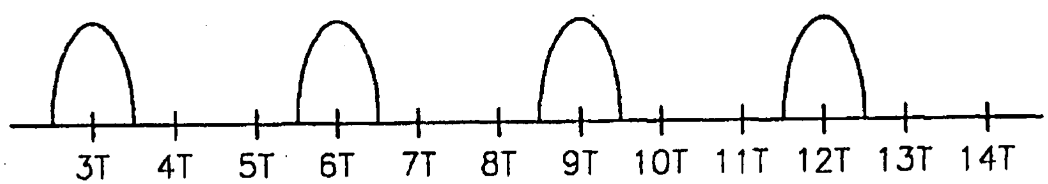


FIG. 4

CDS	CODEWORDS
1	53
−1	0
3	48
−3	50
5	0
−5	41
7	33
−7	0
9	19
−9	25
11	0
−11	12
13	7
−13	0
15	2
−15	3
TOTAL	293

FIG. 5A

DATA	CODEWORD	DATA	CODEWORD
000	000000010000000010000000010	064	000010000010010010000010010
001	000000010000000010000010000	065	000010000010010010010000010
002	000000010000000010000010010	066	000010000010010010010010000
003	000000010000000010010000010	067	000010000010010010010010010
004	000000010000000010010010000	068	000010010000000010000000010
005	000000010000000010010010010	069	000010010000000010000010000
006	000000010000010000000010000	070	000010010000000010000010010
007	000000010000010000000010010	071	000010010000000010010000010
008	000000010000010000010000010	072	000010010000000010010010000
009	000000010000010000010010000	073	000010010000000010010010010
010	000000010000010000010010010	074	000010010000010000000010000
011	000000010000010010000000010	075	000010010000010000000010010
012	000000010000010010000010000	076	000010010000010000010000010
013	000000010000010010000010010	077	000010010000010000010010000
014	000000010000010010010000010	078	000010010000010000010010010
015	000000010000010010010010000	079	000010010000010010000000010
016	000000010000010010010010010	080	000010010000010010000010000
017	000000010010000000010000010	081	000010010000010010000010010
018	000000010010000000010010000	082	000010010000010010010000010
019	000000010010000000010010010	083	000010010000010010010010000
020	000000010010000010000000010	084	000010010000010010010010010
021	000000010010000010000010000	085	000010010010000000010000010
022	000000010010000010000010010	086	000010010010000000010010000
023	000000010010000010010000010	087	000010010010000000010010010
024	000000010010000010010010000	088	000010010010000010000000010
025	000000010010000010010010010	089	000010010010000010000010000
026	000000010010010000000010000	090	000010010010000010000010010
027	000000010010010000000010010	091	000010010010000010010000010
028	000000010010010000010000010	092	000010010010000010010010000
029	000000010010010000010010000	093	000010010010000010010010010
030	000000010010010000010010010	094	000010010010010000000010000
031	000000010010010010000000010	095	000010010010010000000010010
032	000000010010010010000010000	096	000010010010010000010000010
033	000000010010010010000010010	097	000010010010010000010010000
034	000000010010010010010000010	098	000010010010010000010010010
035	000000010010010010010010000	099	000010010010010010000000010
036	000000010010010010010010010	100	000010010010010010000010000
037	000010000000000010000010000	101	000010010010010010000010010
038	000010000000000010000010010	102	000010010010010010010000010
039	000010000000000010010000010	103	000010010010010010010010000
040	000010000000000010010010000	104	000010010010010010010010010
041	000010000000000010010010010	105	010000000010000000010000010
042	000010000000010010000000010	106	010000000010000000010010000
043	000010000000010010000010000	107	010000000010000000010010010
044	000010000000010010000010010	108	010000000010000010000000010
045	000010000000010010010000010	109	010000000010000010000010000
046	000010000000010010010010000	110	010000000010000010000010010
047	000010000000010010010010010	111	010000000010000010010000010
048	000010000010000000010000010	112	010000000010000010010010000
049	000010000010000000010010000	113	010000000010000010010010010
050	000010000010000000010010010	114	010000000010010000000010000
051	000010000010000010000000010	115	010000000010010000000010010
052	000010000010000010000010000	116	010000000010010000010000010
053	000010000010000010000010010	117	010000000010010000010010000
054	000010000010000010010000010	118	010000000010010000010010010
055	000010000010000010010010000	119	010000000010010010000000010
056	000010000010000010010010010	120	010000000010010010000010000
057	000010000010010000000010000	121	010000000010010010000010010
058	000010000010010000000010010	122	010000000010010010010000010
059	000010000010010000010000010	123	010000000010010010010010000
060	000010000010010000010010000	124	010000000010010010010010010
061	000010000010010000010010010	125	010000001000000010000000010
062	000010000010010010000000010	126	010000001000000010000010000
063	000010000010010010000010000	127	010000001000000010000010010

FIG. 5B

DATA	CODEWORD	DATA	CODEWORD
128	010000001000000000100100000010	192	010010000010010010010000010010
129	010000001000000000100100010000	193	010010010000000001000000000010
130	010000001000000000100100010010	194	010010010000000001000000010000
131	010000001000001000000000010000	195	010010010000000001000000010010
132	010000001000001000000000010010	196	010010010000000001001000000010
133	010000001000001000000100000010	197	010010010000000001001000010000
134	010000001000001000000100010000	198	010010010000000001001000010010
135	010000001000001000000100010010	199	010010010000010000000000010000
136	010000001000001001000000000010	200	010010010000010000000000010010
137	010000001000001001000000010000	201	010010010000010000001000000010
138	010000001000001001000000010010	202	010010010000010000001000010000
139	010000001000001001001000000010	203	010010010000010000001000010010
140	010000001000001001001000010000	204	010010010000010010000000000010
141	010000001000001001001000010010	205	010010010000010010000000010000
142	010000001001000000000010000010	206	010010010000010010000000010010
143	010000001001000000000010010000	207	010010010000010010010000000010
144	010000001001000000000010010010	208	010010010000010010010000010000
145	010000001001000000100000000010	209	010010010000010010010000010010
146	010000001001000000100000010000	210	010010010010000000000100000010
147	010000001001000000100000010010	211	010010010010000000000100010000
148	010000001001000000100100000010	212	010010010010000000000100010010
149	010000001001000000100100010000	213	010010010010000001000000000010
150	010000001001000000100100010010	214	010010010010000001000000010000
151	010000001001001000000000010000	215	010010010010000001000000010010
152	010000001001001000000000010010	216	010010010010000001001000000010
153	010000001001001000000100000010	217	010010010010000001001000010000
154	010000001001001000000100010000	218	010010010010000001001000010010
155	010000001001001000000100010010	219	010010010010010000000000010000
156	010000001001001001000000000010	220	010010010010010000000000010010
157	010000001001001001000000010000	221	010010010010010000001000000010
158	010000001001001001000000010010	222	010010010010010000001000010000
159	010000001001001001001000000010	223	010010010010010000001000010010
160	010000001001001001001000010000	224	010010010010010010000000000010
161	010000001001001001001000010010	225	010010010010010010000000010000
162	010010000000001000000000010000	226	010010010010010010000000010010
163	010010000000001000000000010010	227	010010010010010010010000000010
164	010010000000001000000100000010	228	010010010010010010010000010000
165	010010000000001000000100010000	229	010010010010010010010000010010
166	010010000000001000000100010010	230	000000001000000000000010000010
167	010010000000001001000000000010	231	000000001000000000000010010000
168	010010000000001001000000010000	232	000000001000000000000010010010
169	010010000000001001000000010010	233	000000001000000100000000000010
170	010010000000001001001000000010	234	000000001001000000000000010000
171	010010000000001001001000010000	235	000000001001000000000000010010
172	010010000000001001001000010010	236	000000001001001000000000000010
173	010010000010000000000100000010	237	000010000000000000100000000010
174	010010000010000000000100010000	238	000010000000000000100000010000
175	010010000010000000000100010010	239	000010000000000000100000010010
176	010010000010000001000000000010	240	000010000000000000100100000010
177	010010000010000001000000010000	241	000010000000000000100100010000
178	010010000010000001000000010010	242	000010000000000000100100010010
179	010010000010000001001000000010	243	000010000000001000000000000010
180	010010000010000001001000010000	244	000010000010000000000000010000
181	010010000010000001001000010010	245	000010000010000000000000010010
182	010010000010010000000000010000	246	000010000010010000000000000010
183	010010000010010000000000010010	247	000010010000000000000010000010
184	010010000010010000001000000010	248	000010010000000000000010010000
185	010010000010010000001000010000	249	000010010000000000000010010010
186	010010000010010000001000010010	250	000010010000010000000000000010
187	010010000010010010000000000010	251	000010010010000000000000010000
188	010010000010010010000000010000	252	000010010010000000000000010010
189	010010000010010010000000010010	253	000010010010010000000000000010
190	010010000010010010010000000010	254	010000000000001000000000010000
191	010010000010010010010000010000	255	010000000000001000000000010010

FIG. 6A

DATA	CODEWORD	DATA	CODEWORD
000	000000010000000000000010000010	064	000000010010010000000010010010
001	000000010000000000010000010000	065	000000010010010010010010000010
002	000000010000000000010010010010	066	000010000000000000010000010010
003	000000010000000010010000010010	067	000010000000000000010010010000
004	000000010000000010010010010000	068	000010000000010000000000010010
005	000000010010010000000000010010	069	000010000000010000000010000010
006	000000010010010000000010010000	070	000010000000010000000010010000
007	000010000000000000010000000010	071	000010000000010010000000010000
008	000010000000000010000000010000	072	000010000000010010010010010010
009	000010000000000010000010010010	073	000010000010000000000010000010
010	000010000000000010010010000010	074	000010000010000010000000000010
011	000010000000010000010000010010	075	000010000010000010000000010000
012	000010000000010000010010010000	076	000010000010000010010010010010
013	000010000000010010000010000010	077	000010000010010000000000010000
014	000010000000010010010000010000	078	000010000010010000010010010010
015	000010000000010010010010010010	079	000010000010010010000010010010
016	000010010000000010000000010010	080	000010000010010010010000000010
017	000010010000000010000010010000	081	000010000010010010010010010000
018	000010010010000000000010000010	082	000010010000000010000000000010
019	000010010010000010000000010000	083	000010010000010000000000000010
020	000010010010000010010010010010	084	000010010000010000000000010000
021	000010010010010000010000010010	085	000010010000010000000010010010
022	000010010010010010010010010000	086	000010010000010010010000000010
023	010000000000000010000000000010	087	000010010010000000000000010000
024	010000000010000000000000010000	088	000010010010000000000010010010
025	010000000010000000000010010010	089	000010010010000000010000010010
026	010000000010000010010000000010	090	000010010010000000010010000010
027	010000000010010010000000000010	091	000010010010000000010010010000
028	010000010000000000010000010010	092	000010010010010000000010000010
029	010000010000000000010010010000	093	000010010010010010000000000010
030	010000010000010000010000000010	094	000010010010010010000000010000
031	010000010000010010000000010000	095	000010010010010010010010010010
032	010000010000010010010010010010	096	010000000000010000000000010010
033	010000010010000010000000000010	097	010000000000010000000010010000
034	010000010010010000000000010000	098	010000000010000000000000010010
035	010000010010010000000010010010	099	010000000010000000000010000010
036	010000010010010010010000000010	100	010000000010000000000010010000
037	010010000000000010000000010010	101	010000000010000010000000010000
038	010010000000000010000010010000	102	010000000010000010010010010010
039	010010000010000000000010000010	103	010000000010010010000000010010
040	010010000010000010000000010000	104	010000000010010010010010010000
041	010010000010000010010010010010	105	010000010000000000000010000010
042	010010000010010010000000010010	106	010000010000000000010000000010
043	010010000010010010010010010000	107	010000010000000000010000010000
044	010010010000000010000000000010	108	010000010000000000010010010010
045	010010010000010000000000010000	109	010000010000010000000000010000
046	010010010000010000000010010010	110	010000010000010000000010010010
047	010010010000010010010000000010	111	010000010000010010000000010010
048	010010010010000010000000010010	112	010000010000010010010000000010
049	010010010010000010010010010000	113	010000010000010010010010010000
050	010010010010010000010000000010	114	010000010010000010000000010010
051	010010010010010010000000010000	115	010000010010000010000010010000
052	010010010010010010010010010010	116	010000010010010000000000010010
053	000000010000000000000010010010	117	010000010010010000000010000010
054	000000010000000000010000010010	118	010000010010010000000010010000
055	000000010000000000010010000010	119	010000010010010010000000010000
056	000000010000000000010010010000	120	010000010010010010010010010010
057	000000010000010000000010000010	121	010010000000000000010000000010
058	000000010000010010000000000010	122	010010000000000010000000000010
059	000000010000010010000000010000	123	010010000000000010000000010000
060	000000010000010010010010010010	124	010010000000000010000010010010
061	000000010010000010000000000010	125	010010000000000010010010000010
062	000000010010010000000000000010	126	010010000010000000000000010000
063	000000010010010000000000010000	127	010010000010000000000010010010

FIG. 6B

DATA	CODEWORD	DATA	CODEWORD
128	010010000010000010000010010	192	000000010000000000010010000
129	010010000010000010010000010	193	000010000000000010000010000
130	010010000010000010010010000	194	000010000000000010010010010
131	010010000010010000010000010	195	000010000000010010000010010
132	010010000010010010000000010	196	000010000000010010010010000
133	010010000010010010000010000	197	000010000010010000000010010
134	010010000010010010010010010	198	000010000010010000010010000
135	010010010000000010000010010	199	000010010010000000000010010
136	010010010000000010010010000	200	000010010010000000010010000
137	010010010000010000000010010	201	010000000000010000000010000
138	010010010000010000010000010	202	010000000000010000010010010
139	010010010000010000010010000	203	010000000000010010010000010
140	010010010000010010000010000	204	010000000010000010000010010
141	010010010000010010010010010	205	010000000010000010010010000
142	010010010010000000010000010	206	010000000010010000010000010
143	010010010010000010000000010	207	010000000010010010000010000
144	010010010010000010000010000	208	010000000010010010010010010
145	010010010010000010010010010	209	010000010000010000000010010
146	010010010010010000000010000	210	010000010000010000010010000
147	010010010010010000010010010	211	010000010010000000010000010
148	010010010010010010000010010	212	010000010010000010000010000
149	010010010010010010010000010	213	010000010010000010010010010
150	010010010010010010010010000	214	010000010010010010000010010
151	000000010000000010000000010	215	010000010010010010010010000
152	000000010000010000000010000	216	010010000010000000000010010
153	000000010000010000010010010	217	010010000010000000010010000
154	000000010000010010010000010	218	010010010000000000010000010
155	000000010010000010000010010	219	010010010000000010000010000
156	000000010010000010010010000	220	010010010000000010010010010
157	000000010010010000010000010	221	010010010000010010000010010
158	000000010010010010000010000	222	010010010000010010010010000
159	000000010010010010010010010	223	010010010010010000000010010
160	000010000000010000000000010	224	010010010010010000010010000
161	000010000010000000000010000	225	000000010000010000000010010
162	000010000010000000010010010	226	000000010000010000010010000
163	000010000010000010010000010	227	000000010010000000010000010
164	000010000010010010000000010	228	000000010010000010000010000
165	000010010000000010000010010	229	000000010010000010010010010
166	000010010000000010010010000	230	000000010010010010000010010
167	000010010000010000010000010	231	000000010010010010010010000
168	000010010000010010000010000	232	000010000000000010010000010
169	000010010000010010010010010	233	000010000000010010000000010
170	000010010010000010000000010	234	000010000010000000000010010
171	000010010010010000000010000	235	000010000010000000010010000
172	000010010010010000010010010	236	000010000010010000000000010
173	000010010010010010010000010	237	000010010000000000010000010
174	010000010000000000010010010	238	000010010000000010000010000
175	010000010000000010010000010	239	000010010000000010010010010
176	010000010000010010000000010	240	000010010000010010000010010
177	010000010010010000000000010	241	000010010000010010010010000
178	010010000000000010000010010	242	000010010010010000000010010
179	010010000000000010010010000	243	000010010010010000010010000
180	010010000000010000010000010	244	010000000000010000010000010
181	010010000000010010000010000	245	010000000000010010000010000
182	010010000000010010010010010	246	010000000000010010010010010
183	010010000010000010000000010	247	010000000010000010000000010
184	010010000010010000000010000	248	010000000010010000000010000
185	010010000010010000010010010	249	010000000010010000010010010
186	010010000010010010010000010	250	010000000010010010010000010
187	010010010000010000000000010	251	010000010000000000010010000
188	010010010010000000000010000	252	010000010000010000000000010
189	010010010010000000010010010	253	010000010010000000000010000
190	010010010010000010010000010	254	010000010010000000010010010
191	010010010010010010000000010	255	010000010010000010010000010

FIG. 7

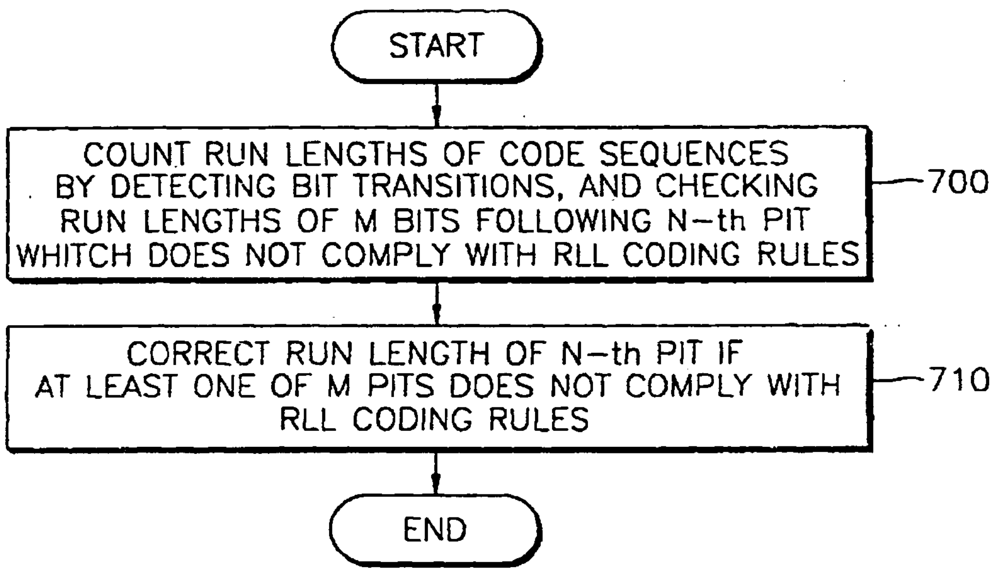

START
COUNT RUN LENGTHS OF CODE SEQUENCES BY DETECTING BIT TRANSITIONS, AND CHECKING RUN LENGTHS OF M BITS FOLLOWING N-th PIT WHITCH DOES NOT COMPLY WITH RLL CODING RULES
700
CORRECT RUN LENGTH OF N-th PIT IF AT LEAST ONE OF M PITS DOES NOT COMPLY WITH RLL CODING RULES
710
END

FIG. 8

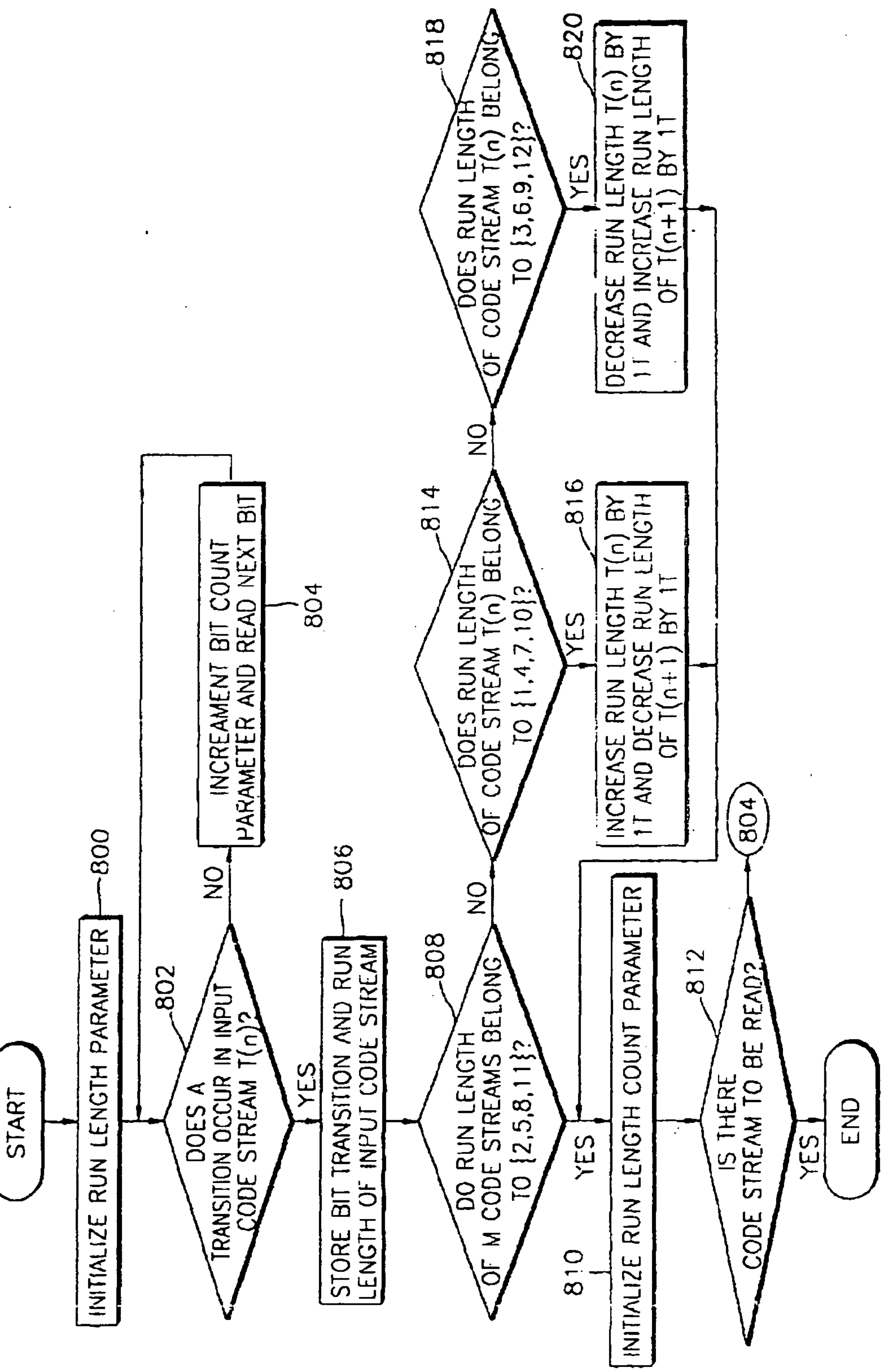

START
INITIALIZE RUN LENGTH PARAMETER
800
802
DOES A TRANSITION OCCUR IN INPUT CODE STREAM T(n)?
NO
YES
INCREAMENT BIT COUNT PARAMETER AND READ NEXT BIT
804
STORE BIT TRANSITION AND RUN LENGTH OF INPUT CODE STREAM
806
808
DO RUN LENGTH OF M CODE STREAMS BELONG TO {2,5,8,11}?
NO
YES
814
DOES RUN LENGTH OF CODE STREAM T(n) BELONG TO {1,4,7,10}?
NO
YES
816
INCREASE RUN LENGTH T(n) BY 1T AND DECREASE RUN LENGTH OF T(n+1) BY 1T
818
DOES RUN LENGTH OF CODE STREAM T(n) BELONG TO {3,6,9,12}?
YES
820
DECREASE RUN LENGTH T(n) BY 1T AND INCREASE RUN LENGTH OF T(n+1) BY 1T
810
INITIALIZE RUN LENGTH COUNT PARAMETER
812
IS THERE CODE STREAM TO BE READ?
804
YES
END

FIG. 9

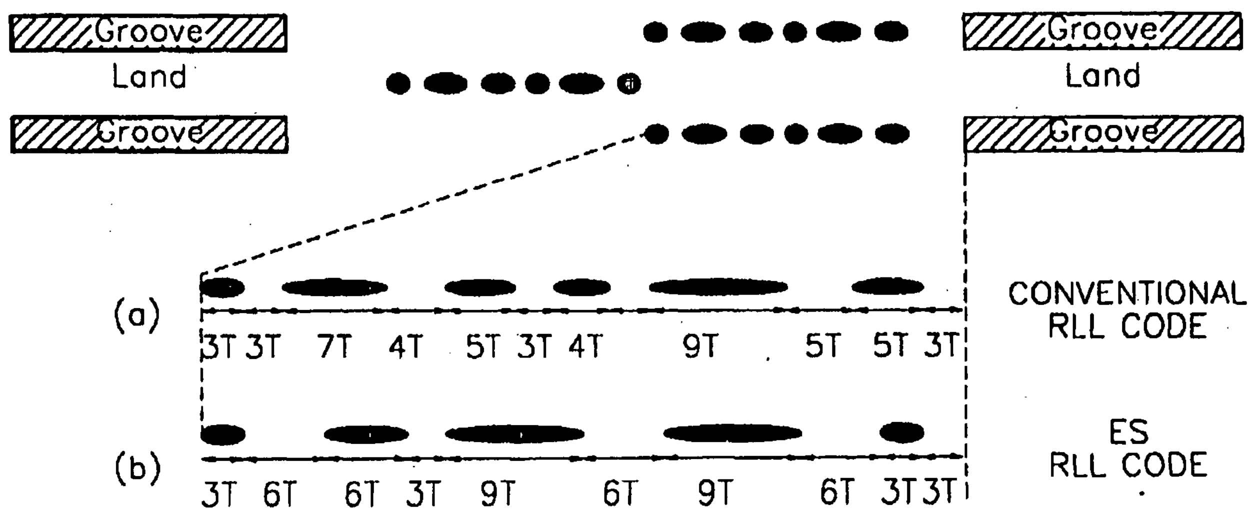


FIG. 10

SEGMENT	VFO	AM	PID	IED
BYTES	21	2	4	2
BITS	576	54	108	54
BITS IN TOTAL	783			

FIG. 11A (PRIOR ART)

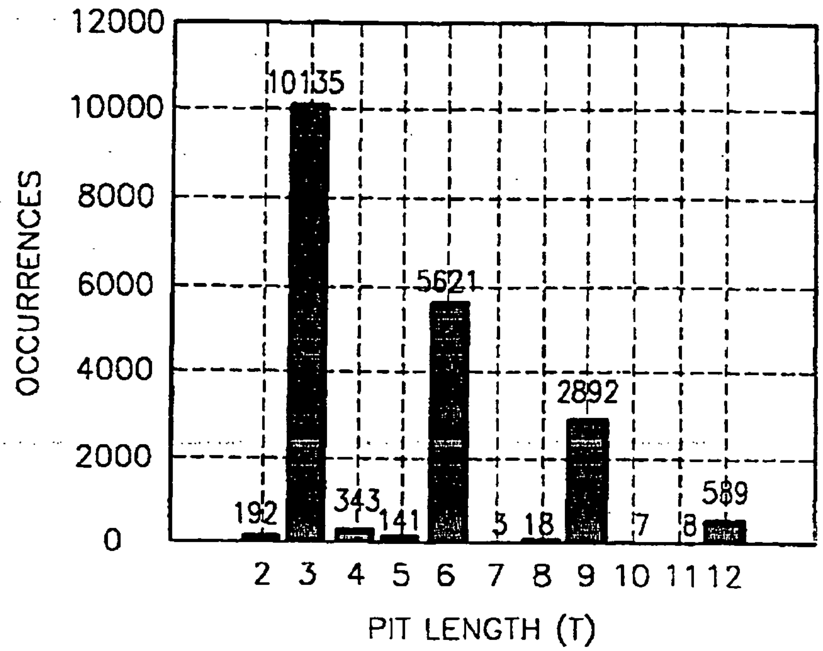


FIG. 11B

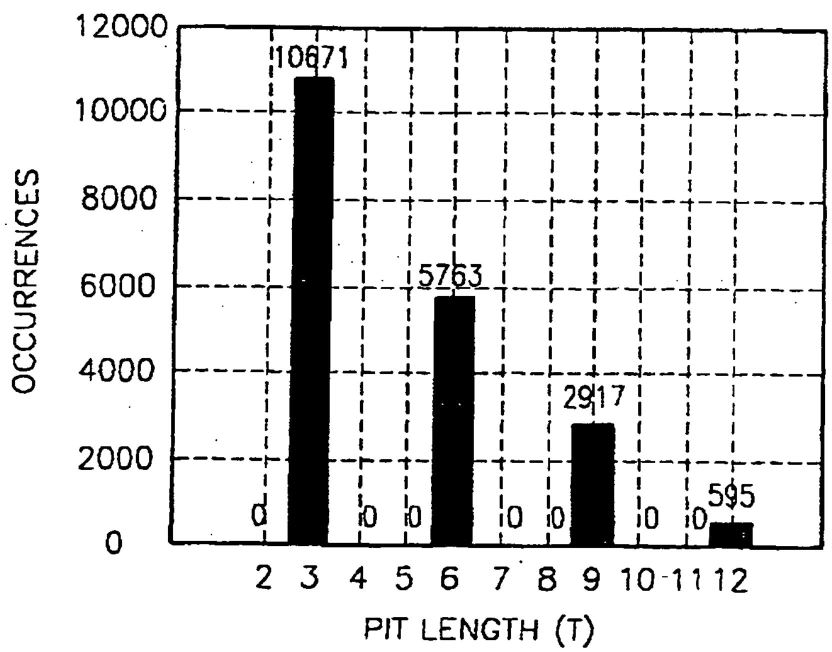


FIG. 12

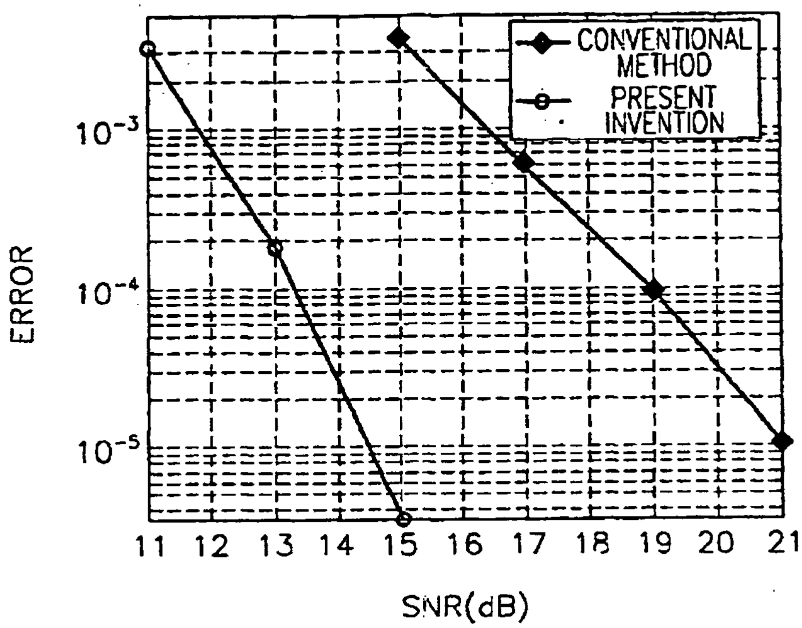

CONVENTIONAL METHOD
PRESENT INVENTION
ERROR
10^{-3}
10^{-4}
10^{-5}
11 12 13 14 15 16 17 18 19 20 21
SNR(dB)

FIG. 13

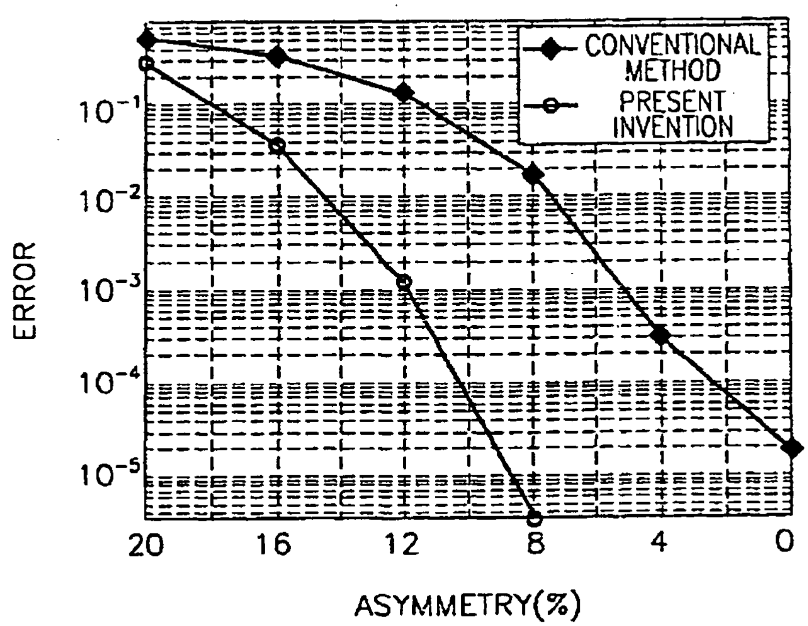
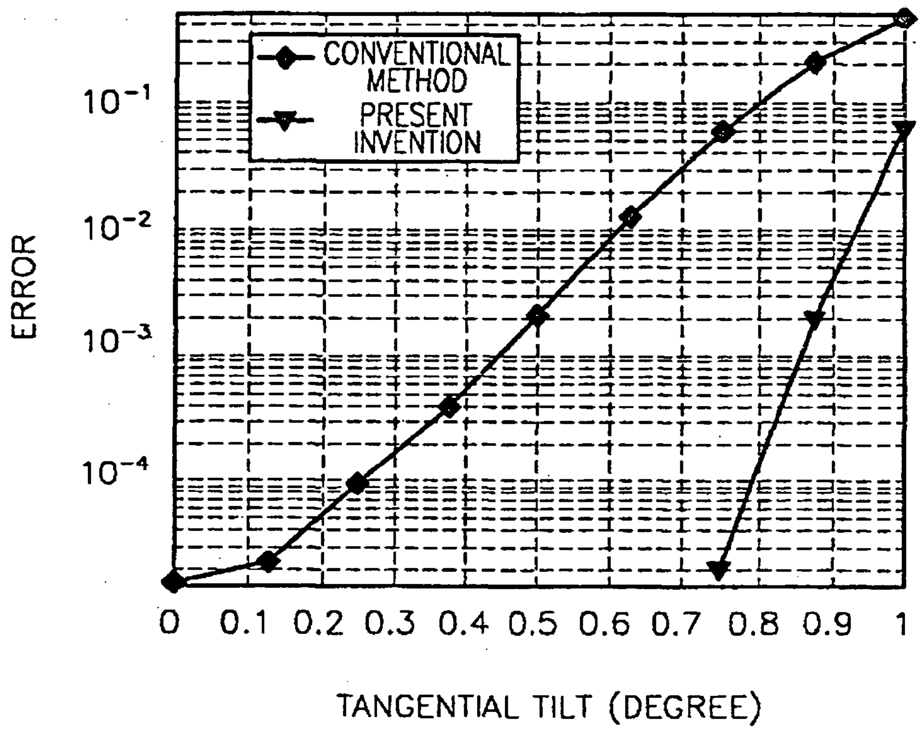

CONVENTIONAL METHOD
PRESENT INVENTION
ERROR
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
20 16 12 8 4 0
ASYMMETRY(%)

FIG. 14


CONVENTIONAL METHOD
PRESENT INVENTION
10^{-1}
10^{-2}
10^{-3}
10^{-4}
ERROR
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
TANGENTIAL TILT (DEGREE)